2006 IEEE Conference on
Systems, Man, and Cybernetics
October 8-11, 2006, Taipei, Taiwan

New Trends in Intelligent Electronic Navigational Aids

Costin Barbu, Maura Lohrenz and Geary Layne

***Abstract*— The smart management of clutter is a key component in designing intelligent, next-generation user interfaces and electronic displays. Intelligent devices can enhance a user's situational awareness under adverse conditions. In this paper we present two approaches to assist a user with target detection and clutter analysis, and we suggest how these tools could be integrated with an electronic chart system. The first tool, an information fusion technique, is a multiple-view generalization of AdaBoost, which can assist a user in finding a target partially obscured by display clutter. The second technique clusters geospatial features on an electronic display and determines a meaningful measure of display clutter. The clutter metric correlates with preliminary, subjective, clutter rankings. The metric can be used to warn a user if display clutter is a potential hazard for his performance. We compare the performance of the proposed techniques with recent classifier fusion strategies on a set of synthetic data.**

I. INTRODUCTION

Over fifteen years ago, the US Navy first installed moving-map displays in the F/A-18 Hornet and AV-8B Harrier aircraft. Electronic charts are now commonplace in military and commercial aircraft, surface ships, and automobiles, and have proven essential to anyone needing immediate access to up-to-date geospatial information, such as meteorologists and air traffic controllers. As new sources of information become available for display, and as new and innovative display techniques are developed, there is a tendency to display everything that might be of interest to the user. These new displays introduce potential human factors' issues with regard to the ability of the user to access and interpret the displayed information. Many studies have linked display complexity to user performance; e.g., display clutter has been shown to significantly disrupt a pilot's visual attention, resulting in greater uncertainty concerning target locations [1], [18], [19]. When a moving-map scrolls at a high rate of speed, as in a fighter jet's cockpit display, the chart's effectiveness can decrease substantially. While researchers have demonstrated a link between user performance and the presence of so-called "clutter" (which can include both the overcrowding of otherwise important information as well as unwanted data or noise), we still lack a reliable method of automatically quantifying display clutter in a way that can be empirically tied to performance.

We illustrate the concept of information fusion employed by the first tool via a simple example.

Given a set of training points $X = \{x_1, x_2, ..., x_N\}$ and M disjoint features available for each point

$$x_i = \{x_i^1, x_i^2, ..., x_i^M\} \quad (1)$$

Each member x_i^j in the set x_i is known as a *view* of point x_i. A *view* may be thought of as a representation of point x_i using disjoint feature sets. For instance, in a color image, each training point x_i may be thought of as a set of three views, each of which consists of one of the three disjoint features obtained from the intensities of Red, Green and Blue color components. In this case, the number of views will be three, represented as $\{x_i^R, x_i^G, x_i^B\}$. Similarly, for a moving target captured using visible range and infrared sensors, the number of views available for each training point in the training set will be two.

The goal of classifier fusion is to obtain a classifier C such that C learns from all the views available for each training point and has classification accuracy that is better than the case when only one view is available. One can ask how helpful could introducing additional views be? A toy example can be used to illustrate this concept. In Fig. 1 (a and b), two classes (circles and squares) are displayed on the OX and OY axes. It is not always possible to separate the classes using information from a single view. On the other hand, if information from all the views is combined, a better classification performance may be achieved. It is generally known that a good fusion algorithm outperforms or at least performs as well as the individual classifiers [14]. Considerable research in the pattern recognition field is focused on fusion rules that aggregate the outputs of the first level experts and make a final decision. Various techniques for fusion of expert observations such as linear weighted voting, the naive Bayes classifiers, the kernel function approach, potential functions, decision trees or multilayer perceptrons have been proposed in recent years, [9]. Other approaches are based on bagging, boosting, and arching classifiers [4], [5]. Comprehensive surveys of various classifier fusion studies and approached can be found in [10] and [11]. In [11] various classifier fusion strategies such as minimum, maximum, average, majority vote and oracle are discussed and the results have been compared. Kuncheva et al. [12] discuss the effect of dependence between individual classifiers in classifier fusion. They study the limits on the majority vote accuracy when combining dependent classifiers. A Q statistics based measure has been proposed to quantify the dependence between the classifiers. It is shown that dependent classifiers could offer a dramatic improvement over the individual accuracy. A synthetic experiment demonstrates the intuitive result that, in general, negative dependence is preferable. In [20] Wolpert proposes

C. Barbu, M. Lohrenz and G. Layne are with Naval Research Laboratory, Stennis Space Center, MS, USA {cbarbu,mlohrenz,glayne}@nrlssc.navy.mil

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
OCT 2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
New Trends in Intelligent Electronic Navigational Aids	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Naval Research Laboratory,1005 Balch Blvd,Stennis Space Center ,MS,39529**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
2006 IEEE Conference on Systems, Man, and Cybernetics, Oct 8-11, 2006, Taipei, Taiwan

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**6**	

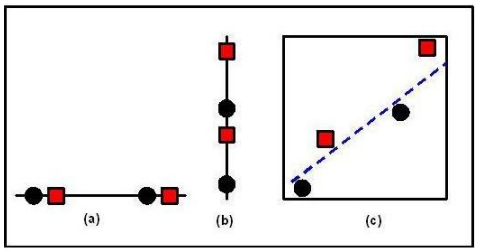

Fig. 1. Analogy between using additional information from two different views and using information from two different dimensions

stacked generalization, a general technique for construction of multi-level learning systems. In the context of classifier combination, it yields unbiased, full-size training sets for the trainable combiner. He defines stacked generalization as any scheme that feeds the information from one set of classifiers (generalizers) to another before forming the final opinion. Lanckriet et al. introduce in [13] a kernel-based data fusion approach for protein function prediction in yeast. The method presented in that paper combines multiple kernel representations in an optimal fashion by formulating the problem as a convex optimization problem that can be solved using semidefinite programming techniques.

In this paper, we present two tools that can be integrated with Intelligent Electronic Navigational Devices such that a user can be assisted when display clutter disrupts his visual attention. The first tool, a classifier fusion technique, is detailed in [3] and [2]. For the sake of clarity that fusion algorithm is briefly described in the next section. The second tool is a feature clustering-based technique that analyzes the display clutter. Based on that analysis, the user can be warned by visual or acoustical alarm signals that his performance can be affected by the lack of the chart's effectiveness.

II. BOOSTING DISJOINT VIEWS USING SHARED SAMPLING DISTRIBUTION

AdaBoost has been shown to improve the prediction accuracy of weak classifiers using an iterative weight update process [5]. The technique combines weak classifiers (classifiers having classification accuracy slightly greater than that of chance) in a weighted vote fashion giving an overall strong classifier. Detailed explanation of the AdaBoost algorithm is skipped here for brevity, interested readers may refer to [16], [17] and [6] for more on AdaBoost. One of the ways boosting may be used for classifier fusion would be to run boosting separately on each view, obtain separate ensembles for each view and take a majority vote among the ensembles when presented with test data. In this case, separate training of classifiers is needed for each view and the sampling distributions of the data points are also disjoint. Unlike this approach, we perform separate training for each view but the training error computation and sampling of training examples is done using a shared distribution of example weights in a given iteration. The training algorithm is shown in Algorithm 1.

Algorithm1 :Boosting with Shared Sampling Distribution (BSSD)

Input:

1. N training examples in a training set S.
2. M views available for each training point and hence M training sets such that $S_j = \langle (x_1^j, y_1), (x_2^j, y_2) ..... (x_N^j, y_N) \rangle$ where $j = 1....M$ and $y_i \in \{+1, -1\}$ and each (x_i^j, y_i) pair represents the j^{th} view and class label of the i^{th} training example.

Initialization: The weights of the training examples are initialized to $w_1(i) = \frac{1}{N}$.

For $k = 1$ **to** k_{max}

1. For each view j, train classifiers C_k^j, using W_k
2. Obtain weak hypotheses h_k^j, for each view j
3. Obtain the error rates ε_k^j of each h_k^j over the distribution W_k such that $\varepsilon_k^j = P_{i \sim W_k}[h_k^j(x_i^j) \neq y_i]$
4. If errors from each of the M views, $\{\varepsilon_k^1, \varepsilon_k^2 ..... \varepsilon_k^M\} < 0.5$, select h_k^* with the lowest error rate ε_k^* amongst all views
5. Compute the value $\alpha_k^* = \frac{1}{2} ln(\frac{1-\varepsilon_k^*}{\varepsilon_k^*})$ where $\varepsilon_k^* = \min\{\varepsilon_k^1, \varepsilon_k^2 ..... \varepsilon_k^M\}$ and α_k^* is the corresponding combination weight value.
6. Update the weights

$$w_{k+1}(i) = \frac{w_k(i)}{Z_k^*} \times \begin{cases} e^{-\alpha_k^*} & \text{if } h_k^*(x_i^*) = y_i \\ e^{\alpha_k^*} & h_k^*(x_i^*) \neq y_i \end{cases}$$

where h_k^* is the classifier with lowest error rate ε_k^* in the k^{th} iteration. Z_k^* is the normalizing factor so that W_{k+1} is a distribution

Output: $F(x) = \sum_{k=1}^{k_{max}} \alpha_k^* h_k^*(x^*)$

$Final\ hypothesis: H(x) = sign(F(x))$

In the initialization step of Algorithm 1, all the views for a given training point are initialized with the same weight. To understand this we go back to the RGB component example. Suppose we have N training examples each having three disjoint views such that a given training example x_i can be represented as $x_i = \{x_i^R, x_i^G, x_i^B\}$. Weak learners h^R, h^G and h^B will be trained on the training sets $X^R = \{x_1^R, x_2^R ..... x_N^R\}$, $X^G = \{x_1^G, x_2^G ..... x_N^G\}$ and $X^B = \{x_1^B, x_2^B ..... x_N^B\}$ such that $X = \{X^R \bigcup X^G \bigcup X^B\}$. Since the sampling distribution for all views of a given example is shared, the sampling weight of the the R, G and B views of example x_i in iteration k are given by

$$w_k^{R,G,B}(i) = w_k^R(i) = w_k^G(i) = w_k^B(i).$$

After a classifier h_k^* with lowest error rate ε_k^* is selected in step 4 of Algorithm 1 and combination weight α_k^* is obtained, the weights of the views are updated.

III. QUANTIFYING VISUAL CLUTTER VIA FEATURE CLUSTERING

Previous studies on clutter (e.g. [15] and [21]) focus primarily on the contribution of saliency to image clutter.

We theorize our perception of clutter is related to both saliency and color uniformity, or "density". Saliency refers to how clearly one color or feature "pops out" from the surrounding features in an image, which we estimate by a weighted average of color gradients between adjacent features. Color uniformity refers to how densely-packed are similarly-colored pixels within the image. To calculate this value, we have adapted a clustering algorithm, which we originally developed to cluster seafloor objects detected in sidescan sonar imagery. The algorithm clusters features detected within a predetermined geospatial distance from each other, produces vertices for a bounding cluster polygon, and calculates the cluster's density as the number of clustered features divided by the area of the polygon. For this project, we adapted the clustering algorithm to operate in three-dimensional (3D) space, in which the third dimension is color. Our "color uniformity" value is then derived from the density of similarly-colored pixels within a 3D cluster (i.e., density = a weighted number of points within the cluster divided by the cluster's volume). We describe image clutter in terms of both local and global clutter components. A Local Clutter Metric (LCM) represents the contribution of one color or feature to the overall image clutter, and equals 1 minus the weighted average (by area) of the densities of all clusters centered on that color or feature. A Global Clutter Metric (GCM) represents the overall image clutter, equal to the weighted average of the LCM's for all colors or features in the image. Fig. 2 illustrates our proposed clutter function, in terms of saliency and LCM/GCM. The following sections describe in more detail how each of these metrics is calculated.

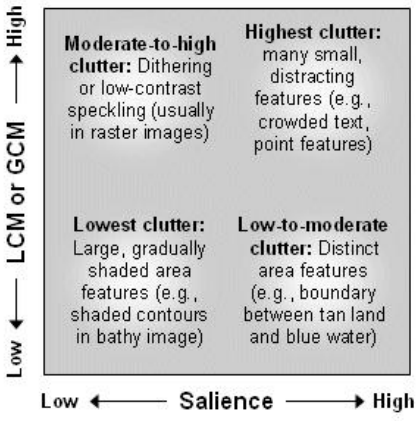

Fig. 2. Clutter as a function of saliency and LCM (for local clutter) or GCM (global clutter)

A. 3D Clustering using Geospatial Bitmaps (GB)

The original clustering algorithm relies on a geospatial bitmapping (GB) technique patented by NRL in 2001 [8]. The algorithm is unique in that it is an autonomous, consistently repeatable, computationally efficient "single-pass" method operating on a user-defined area of interest [7]. The algorithm clusters features by geospatial location and calculates a numerical measure of "cluster density" that considers the number and size of objects clustered in a given area, as well as the scale or resolution of the complete dataset. An enhancement to the original algorithm for this project is the ability to cluster features in three or more dimensions: two geospatial (x, y) dimensions plus a third (z) dimension such as color, size, or feature type. This paper presents preliminary results of clustering by geospatial location and color.

The GB clustering algorithm is a nonhierarchical algorithm with results similar to Nearest Neighbor (NN). NN iteratively calculates and compares the distances between every pair of elements in the dataset to determine which elements should be clustered together. In contrast, the GB algorithm is non-iterative, faster, less computationally intensive, and requires less computer memory than NN. The authors suggest that the GB algorithm is well suited to autonomous clustering applications, because the ordering of elements input to the algorithm has no effect on the resulting clusters (unlike NN and other single-pass methods), and the GB algorithm does not require a seed point to initiate clustering (unlike K-means and other relocation methods). The GB algorithm uses simple bitmaps, in which bits are turned on (set = 1) or off (cleared = 0), indicating the presence or absence of elements of interest. The index of each bit is unique and denotes its position relative to the other bits in the bitmap. In a 2D bitmap, each bit is indexed by its column (x) and row (y); in 3D, each bit is indexed by x, y, and depth (z). Although a GB can be defined for an entire finite space, memory is only allocated - dynamically - when groups of spatially close bits are set, resulting in a compact data structure that supports very fast Boolean and morphological operations. For this project, 3D bitmaps were used to cluster the pixels in an image of interest, based on geospatial location (x, y) and color (z). A separate clustering was performed for each color in the image. For example, Fig. 3 illustrates the results of clustering the shoreline pixels (darker brown color) in the sample image (left). All pixels within a geospatial distance of 1 (x and y) and a color distance of 9 (using the Commission Internationale d'Eclairage (CIE) L*a*b* color space) are included in the clusters (right). In this case, the resulting clusters only contain the shoreline pixels themselves. If z were increased to 10, every pixel in this image would be contained in a single cluster, because every pixel in this image is immediately surrounded by pixels that are within a color distance of 10 in CIE L*a*b* space.

B. Calculating Cluster Density

After clustering all pixels in the image into bounded polygons for a given "seed color" s, a cluster density D_P is calculated for each cluster polygon P:

$$D_P = \frac{\sum(W_c N_c)}{A_P}$$

where: $W_c = Weighting\ factor\ for\ color$ **c** $= 1 - \frac{E_c}{M}$

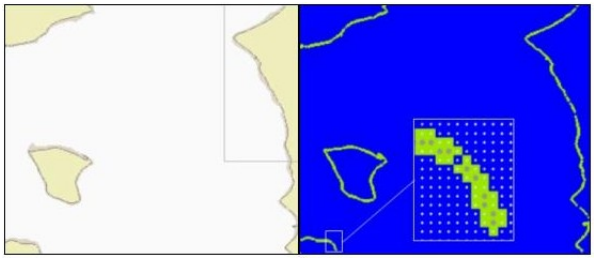

Fig. 3. Example of clustering by geospatial location and color: all pixels within a predetermined distance in geospatial (x=1, y=1) and color (z=9) space of the shoreline pixels (brown pixels in the original image, left) are clustered together. The resulting clusters are shown at right. The zoomed-in section shows a detail of the clustered pixels.

E_c = *Euclidean distance between colors* **c** *and* **s** *in the chosen color space; e.g., for CIE* $L*a*b$:
$= \sqrt{[(L_c - L_s)^2 + (a_c - a_s)^2 + (b_c - b_s)^2]}$
M = *Maximum distance between colors in chosen color space*
N_c = *Number of pixels of color* **c** *in the cluster polygon*
A_P = *Area of cluster polygon* **P**

The color of each pixel in the cluster will be within a color distance of z from all immediately surrounding pixels in the cluster, starting with pixels of color s. In other words, the cluster will "chain" pixels together to form the cluster, starting with each pixel of color s and subsequently including all other pixels within a geospatial distance of x, y and a color distance of z. If $z = 0$, then $D_P = \frac{N_S}{A_P}$. Note the inverse relationship between clutter and density as it is used here: higher density predicts lower clutter, since density describes how closely-packed like-pixels are in the image.

C. Local and Global Clutter Metrics

Local density (D_S) estimates how much an individual seed color (s) contributes to the overall clutter of the image. D_S is computed as the weighted average of the densities for all clusters centered on color s:

$$D_S = \frac{\sum(D_P A_P)}{A_S}$$

where: D_P = *Density of cluster p (described in the previous section)*
A_S = *Sum of areas of all clusters for color* **s**

Global density (D_I), which estimates clutter for the entire image, is computed as the weighted average of the local clutter densities for all colors in the image:

$$D_I = \frac{\sum(D_S A_S)}{A_I}$$

where: D_S = *Weighted average of clutter densities for all clusters centered on color* **s** (*described above*)
A_I = *Sum of all areas* A_S *for image I.*

D. Saliency

We estimate the local saliency of a given color or feature as a weighted average of the color differences between each color or feature of interest and immediately adjacent colors or features. For example, if one feature in the image (e.g., a yellow lighthouse symbol on a nautical chart) is completely surrounded by another feature (e.g., solid blue water), we would estimate the saliency of the lighthouse as the Euclidean distance between these two colors (yellow and blue) in a perceptually representative color space. If this lighthouse symbol were placed on a shoreline (brown), such that 40% of the lighthouse symbol was bordered by the blue water, 40% by tan land, and 20% by the brown shoreline, we would estimate the saliency of the lighthouse by $0.4 * (blue - yellow) + 0.4 * (tan - yellow) + 0.2 * (brown - yellow)$. Global saliency is estimated as the weighted average of the local saliencies for all colors (or features) in the image. Greater color distances result in greater saliency.

The choice of an appropriate color space is central to this theory. Unfortunately, no single color space has been shown to perfectly model human visual perception. For this paper, we chose the standard CIE L*a*b* color space, but we continue to search for improved options.

IV. EXPERIMENTAL RESULTS

We employed our fusion algorithm for target/clutter discrimination on a set of binary class synthetic data. We generated 32 target class images such that a HUD (head-up display) symbol, a Bray-style flight path marker is included in each image as in [21]. A set of 32 clutter class images are represented by images that share a common texture pattern. Sample images from both classes are illustrated in Fig. 4. We consider the fusion of three views represented by the principal component projections, edges and wavelet coefficients for each image.

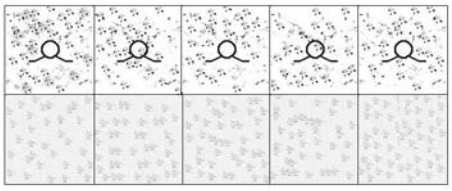

Fig. 4. Sample images of target (first row) and clutter (second row)

We empirically compare BSSD with fusion methods such as stacked generalization (stacking), semidefinite programing (SDP/SVM) and majority vote (SVM–MV). Experimental results are presented in Fig. 6 thru 9 and Tables II thru V. The results represent the average accuracy of 20 tests, each time the data sets being randomly partitioned such that 60% of the data is in the training set and the remaining 40% is in the test set. The average accuracy of individual classifier from each view before fusion is shown in the columns A_{V_1}, A_{V_2} and A_{V_3}. The average fusion accuracy is presented in column A_{fusion}. Naive Bayes classifiers were used as weak learners for boosting.

TABLE I

COMPARISONS OF GCM AND LCM FOR TARGET AND CLUTTER IMAGES

Name	Entire Image		Target color	
	GCM	Saliency	LCM	Saliency
Clutter 1	42.60%	3.7	78.67%	15.0
Clutter 5	38.24%	3.8	79.23%	15.0
Target 14	26.13%	16.7	28.01%	97.0
Target 27	23.76%	7.7	26.87%	98.0

The SVM algorithm used as a back-end generalizer in stacking has two procedural parameters: σ and C, the soft margin parameter. Ten-fold cross-validation was used for model selection, taking σ values in $[10^{-2}, 10^{2}]$ and C in $[10^{-2}, 10^{2}]$. Majority vote is also used for fusion of expert observations for the fusion techniques SVM–MV in which SVM has been used as classifier for each view. We used gaussian, polynomial and linear kernel functions on each view for the semidefinite programming technique. In Fig. 7 thru 9 we compared the robustness of BSSD to noise with the competing techniques. We randomly added various levels of noise to the training data labels on all three views by flipping the labels.

We calculated our global and local clutter metrics for the two sets of images (images with the target symbol surrounded by varying amounts of clutter vs. images with clutter only). Results are presented in Table I and Fig. 5. The local clutter metrics (LCM and saliency) clearly delineated between images containing the target symbol and images containing only clutter. For the clutter-only images, the darkest color of each image was chosen to be the color of interest; for the target images, the target color (black) was chosen. The global metrics did not as clearly distinguish between the images containing the target and those without the target, since both sets of images contained equivalent amounts of background clutter.

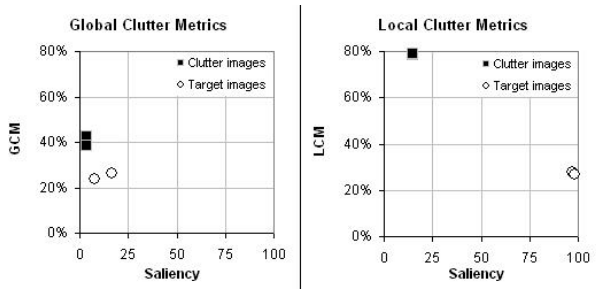

Fig. 5. Plots of Clutter Metrics vs. Saliency for target vs. clutter test images

V. SUMMARY AND DISCUSSION

In this paper we present two tools of potential utility to users of electronic chart displays. The first tool is a boosting-based classifier fusion that can assist a user in finding a target when display clutter disrupts visual attention. The classifier fusion strategy performs classification using weak learners trained on different views of the training data. The final ensemble contains learners trained to focus on different

TABLE II

EXPERIMENTAL RESULTS IN THE ABSENCE OF NOISE

Technique Used	A_{V_1}	A_{V_2}	A_{V_3}	A_{fusion}	Statistical Significance
SVM–MV	0.648	0.544	0.946	0.778	yes
Stacking	0.648	0.544	0.946	0.690	yes
SDP/SVM	*Poly*	*Lin*	*Gaus*	0.482	yes
BSSD	1.000	0.995	0.999	1.000	

TABLE III

EXPERIMENTAL RESULTS IN THE PRESENCE OF NOISE 10%

Technique Used	A_{V_1}	A_{V_2}	A_{V_3}	A_{fusion}	Statistical Significance
SVM–MV	0.642	0.580	0.855	0.763	yes
Stacking	0.642	0.580	0.855	0.598	yes
SDP/SVM	*Poly*	*Lin*	*Gaus*	0.408	yes
BSSD	0.743	0.855	0.995	0.992	

TABLE IV

EXPERIMENTAL RESULTS IN THE PRESENCE OF NOISE 20%

Technique Used	A_{V_1}	A_{V_2}	A_{V_3}	A_{fusion}	Statistical Significance
SVM–MV	0.607	0.525	0.825	0.763	yes
Stacking	0.607	0.525	0.825	0.550	yes
SDP/SVM	*Poly*	*Lin*	*Gaus*	0.482	yes
BSSD	0.679	0.658	0.839	0.992	

TABLE V

EXPERIMENTAL RESULTS IN THE PRESENCE OF NOISE 30%

Technique Used	A_{V_1}	A_{V_2}	A_{V_3}	A_{fusion}	Statistical Significance
SVM–MV	0.542	0.542	0.738	0.675	yes
Stacking	0.542	0.542	0.738	0.559	yes
SDP/SVM	*Poly*	*Lin*	*Gaus*	0.482	yes
BSSD	0.629	0.618	0.743	0.946	

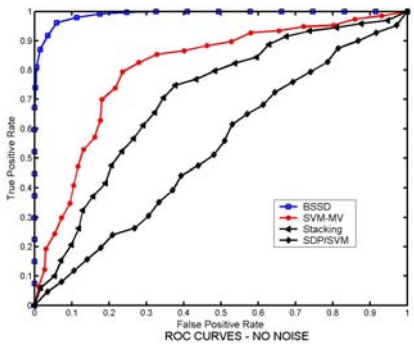

Fig. 6. ROC Curves in the absence of noise

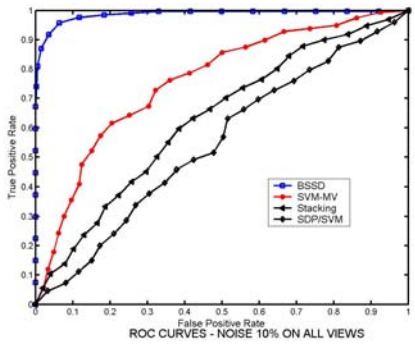

Fig. 7. ROC Curves in the presence of noise 10% on all views

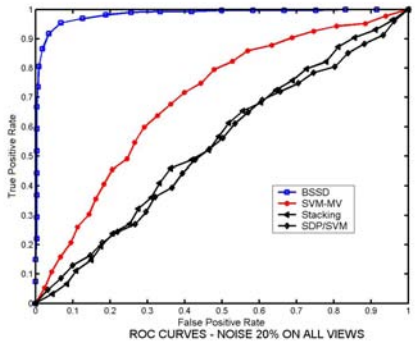

Fig. 8. ROC Curves in the presence of noise 20% on all views

views of the test data. The combination weights for the final weighting rule are obtained using a shared sampling distribution. In each iteration, one weak learner is selected from the pool of weak learners trained on disjoint views. This results in a minimization of the training error for the final hypothesis. It was shown in [3] that a lower training and generalization error bound can be achieved if a shared sampling distribution is used and a weak learner from the lowest error view is selected.

The second tool is a feature clustering technique that analyzes display clutter and attempts to determine whether a target of interest exists. Based on these analyses, the user could be warned by visual or acoustical alarms if his or her performance is likely to be affected by the amount of clutter in the display. The performance of the classifier fusion algorithm has been compared with other data fusion algorithms, namely stacking, majority vote and a semi-definite programming-based kernel method. We show that the proposed technique performs statistically significant better than other fusion techniques with $> 95\%$ confidence using a two-sided paired T-test.

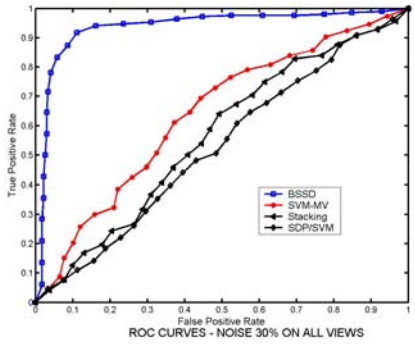

Fig. 9. ROC Curves in the presence of noise 30% on all views

VI. ACKNOWLEDGMENTS

This work was sponsored under Program Element 002435N by the NRL 6.2 Base Program. The authors thank Dr. Marlin Gendron (NRL) for his valued assistance.

REFERENCES

[1] A. Aretz. A model of electronic map interpretation. *Proceedings of the Human Factors Society 32nd Annual Meeting*, pages 130–135, 1988.
[2] C. Barbu. *An Ensemble Approach to Robust Classifier Fusion*. Ph.D. Thesis, Tulane University, New Orleans, 2006.
[3] C. Barbu, R. Iqbal, and J. Peng. Classifier fusion using shared sampling distribution for boosting. *Proceedings of 5th IEEE International Conference on Data Mining*, pages 34–41, 2005.
[4] L. Breiman. Arching classifiers. *Annals of Statistics*, 26:801–849, 1998.
[5] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and Systems Science*, 55:119–139, 1997.
[6] Y. Freund and R. E. Schapire. A short introduction to boosting. *Journal of Japanese Society for Artificial Intelligence*, 5(14):771–780, 1999.
[7] M. Gendron, G. L. M., and M. Lohrenz. A system, method and apparatus for clustering features. *Navy patent case 84,921, Naval Research Laboratory*, 2005.
[8] M. Gendron, P. Wischow, M. Trenchard, M. Lohrenz, L. Riedlinger, and M. Mehaffey. Moving map composer (mmc). *U.S. Patent 6,218,965 B1*, 2001.
[9] S. Hashem. Optimal linear combination of neural networks. *Neural Networks*, 19:599–614, 1997.
[10] J. Kittler. A framework for classifier fusion: Is still needed? *Lecture Notes in Computer Science*, 1876:45–56, 2000.
[11] L. I. Kuncheva. A theoretical study on six classifier fusion strategies. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 24(2):281–286, 1997.
[12] L. I. Kuncheva, C. J. Whitaker, C. A. Shipp, and R. P. W. Duin. Is independence good for combining classifiers? *Proceedings of the 15th International Conference on Pattern Recognition*, 2:168–171, 2000.
[13] G. R. G. Lanckriet, M. H. Deng, N. Cristianini, M. I. Jordan, and W. S. Noble. Kernel-based data fusion and its application to protein function prediction in yeast. *Proceedings of the Pacific Symposium on Biocomputing*, 9:300–311, 2004.
[14] N. S. V. Rao. On fusers that perform better than best sensor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(8):904–909, 2001.
[15] R. Rosenholtz, Y. Li, J. Mansfield, and Z. Jin. Feature congestion: A measure of display clutter. *Proceedings of the Conference on Human Factors in Computing Systems*, 2005.
[16] R. E. Schapire, Y. Freund, P. Bartlett, and W. Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Proceedings of the Fourteenth International Conference on Machine learning*, pages 322–330, 1997.
[17] R. E. Schapire and Y. Singer. Improved boosting algorithms using confidence rated predictions. *Machine Learning*, 3(37):297–336, 1999.
[18] V. Schons and C. Wickens. Visual separation and information access in aircraft display layout. *University of Illinois Institution of Aviation Technical Report ARL-93-7/NASA-A3I-93-1*, 1993.
[19] C. Wickens and C. Carswell. The proximity compatibility principle: its psychological foundation and relevance to display design. *Human Factors*, 37(3):473–494, 1995.
[20] D. H. Wolpert. Stacked generalization. *Neural Networks*, 5:241–259, 1992.
[21] M. Zuschlag. Quantification of visual clutter using a computational model of human perception: an application for head-up displays. *Proceedings of the Human Performance, Situational Awareness and Automation*, 2004.